MINING MART: Combining Case-Based-Reasoning and Multi-Strategy Learning into a Framework to reuse KDD-Application

Jörg-Uwe Kietz, Regina Zücker
Swiss Life, Information Systems Research (CC/ITRD),
CH-8022 Zurich, Switzerland
{Uwe.Kietz,Regina.Zuecker}@swisslife.ch

May 15, 2000

Abstract

One of the most time consuming steps for KDD consists in preparing the source data. On the one hand in real-world applications we have to deal with very heterogeneous data of doubtful quality. On the other hand there is no universal data mining tool available which is suited to handle all the various analysis tasks promising enhanced insight about a company's position in the market, it's customers, it's products etc. As a consequence many different types of data mining algorithms have to be employed with typically very strict and specialized input requirements. In this paper, we propose a case-based-reasoning framework for the KDD-process, which does not automatize pre-processing and mining tool selection, but which should make it much easier to reuse the work done for one KDD-task for another similar one. The integration of multiple machine learning (or data minig) supported pre-processing operations, will make the case-adaption at least partially automatic.

1 Introduction

The entire (information) society is in a somewhat paradox situation: we are starving for knowledge while drowning in data. Traditional approaches fail to release the knowledge from the masses of available data. Two technologies are emerging to reduce the gap:

1. data warehousing and on-line analytical processing (OLAP) of data for the verification of hypotheses and
2. knowledge discovery in databases (KDD) for the discovering of new hypotheses.

Practical experience with these techniques have proven their value. However, it is also apparent that using a data warehouse for decision support or applying tools for knowledge discovery are difficult and time-consuming tasks. Therefore, it is currently still quite difficult to get an admissible return of investment from using them. If we inspect real-world applications of knowledge discovery, we realize that 50 - 80% of the efforts are spent on finding an appropriate transformation of the given data, finding appropriate sampling of the data, and specifying the proper target of data mining, i.e. on pre-processing the data. This is not only a time-consuming task, but also a very demanding one, which requires profound data mining and database know-how. As a result these technologies are not used by the common business people, but only by a few highly skilled power users.

To overcome the shortcomings of the current knowledge discovery process, this paper addresses the following four related objectives:

1. Create a user-friendly data mining environment for the non-expert user.
2. Speed up the discovery process by reducing the number and the complexity of trial and error pre-processing and analysis cycles.
3. Minimize the amount of data that is kept within the data mining environment.
4. Improve the quality of data mining results by improving the quality of data.

As the framework to reach these objectives we propose a combination of case-based reasoning and multi-strategy learning. A case in this framework consists of a pairing of

	Time	Imp.
Business understanding	20%	80%
a) Exploring the problem	10%	15%
b) Exploring the solution	9%	14%
c) Implementation specification	1%	51%
Data preparation & mining	80%	20%
a) Data preparation	60%	15%
b) Data surveying	15%	3%
c) Modeling (data mining)	5%	2%

Table 1: Steps of a KDD-project with time to complete and importance to success

business problems and data to analyse with clever multi-strategy learning supported pre-processing and most suitable analysis methods. In this framework the highly skilled data mining power user is still needed, but only to create new cases, not to redo the same cases all the time. This task is delegated to the end-user, who retrieves the prepared cases, makes some simple adaption, e.g. the selection of another target segment. The reapplication of the integrated multi-strategy learning supported pre-processing operations ensures, that the case is automatically adapted to this new data set. Also the power-user can profit from this environment, as the already prepared cases can provide useful building blocks and entire analysis chains for re-use, which should speed-up the initial creation of new data mining business-cases. The pre-processing cases are not stored extensionally as most common in todays KDD Support Environments (KDDSE), but intensional as transformation specification in the form of meta-data [Staudt *et al.*, 1999b; Staudt *et al.*, 1999a] as e.g. in the modern data warehouse ETL-tools (Extract, Transform and Load) like PowerMart (Informatica, http://www.informatica.com) or Datastage (Ardent, http://www.ardentsoftware.com). This ensures, that the cases could be edited easily, re-applied, and - most importantly to tackle objective 3 - could be compiled into a form executable by a data-base system to execute the data transformation.

2 Business Cases, KDD-projects and Data Warehousing

Ideally, a KDD-project starts with a business case, which could be solved or optimized by analyzing available data. The typical steps of such a project are the one's given in table 1 from [Pyle, 1999]. It should be noted that the most important step is the management agreed specification of how to use the expected mining results as the

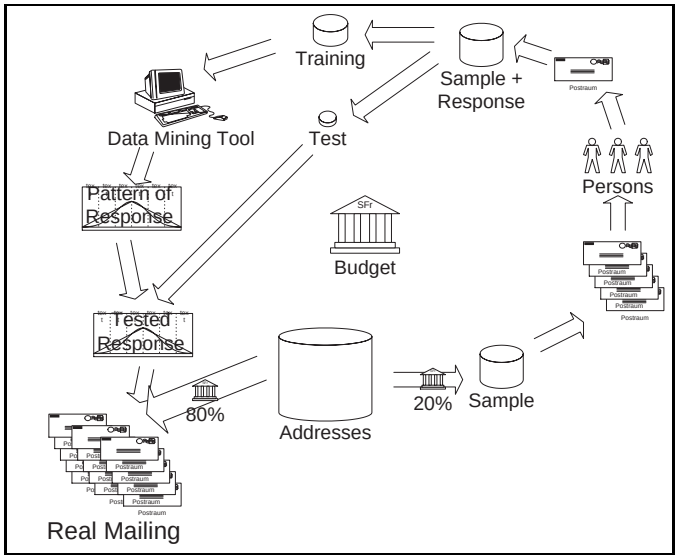

Figure 1: The Business Case Mailing Action

best mining results are worth nothing, if they are not used. The most time-consuming step is the preparation of the data for mining. Both problems could be solved in a related manner. The management support for the use of the mining results as well as the justification of the high data preparation costs are most easily reached, if the KDD-project is integrated into an important and repeated business case. However, the second part is only true, if the pre-processing effort can be reused.

For Swiss Life there are several application areas where central business-cases could be supported by data mining [Staudt *et al.*, 1998], especially:

- Marketing
- Product development and controlling
- Business reporting

In this paper we will use the optimization of responses to mailing actions in direct marketing as an illustrating and well known example of such a problem. This business-case is illustrated in figure 1[1].

If we describe this business-case on a more technical level we come to the following steps:

0. Use an existing data warehouse (DWH) as base.

[1] Ling and Li [1998] describe another problem and solution analysis of this business-case. However their analysis is based on the assumption, that existing customers are persons answering to mailings, which is not the case in our setting, where most contracts are still sold by insurance-agents and not by mailing-actions.

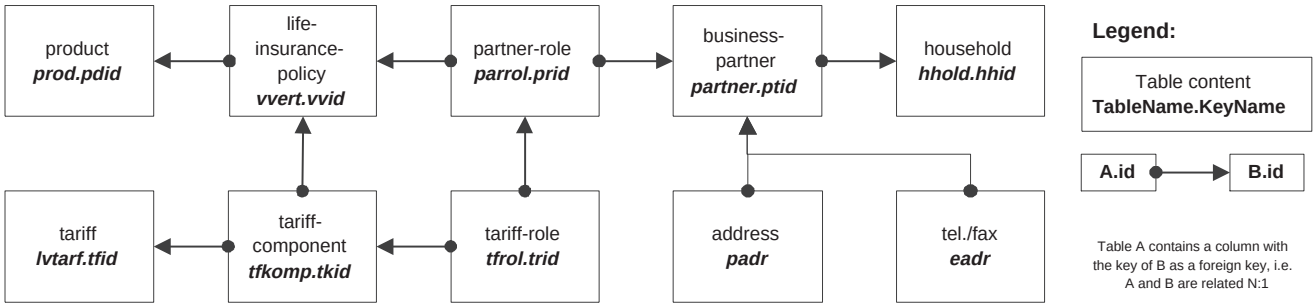

Figure 2: Schema excerpt from our DWH

1. Construct a household view on this DWH, which provides all relevant information in a form, that allows the next step to be done by an end-user.

2. Select the target segment, e.g. households,

   (a) which have a child with an age below 2 and which are not mailed since the birthday of this child, or

   (b) which have already bought a single-premium insurance, but this is more than two years ago, or

   (c) ...

3. select a random-sample, with size proportional[2] to 20% of the budget, i.e. generate a sample to gather labels for the training and test set.

4. Export the addresses of this sample, do the first mailing, and store the responses, i.e. label the sample.

5. Split the sample into training and test-set.

6. Select/Construct the relevant attributes for the current response prediction task.

7. Train the selected mining-tool, which output could be used to order (not just classify) the data.

8. Apply the data-transformations (pre-processing) done in step 6 to the test-data as the mined pattern relies on it.

9. Test the mined pattern on the test-set, i.e get an estimated response-rate for the mined pattern on the target-group.

[2] If your budget is X and a single letter cost Y you can send a total of X/Y letters, spending 20% of the budget to get the training/test data means to select X/Y*0.2 household-records from the target segment.

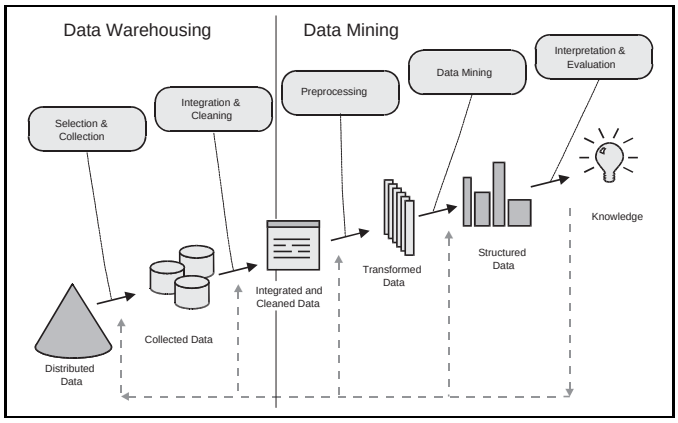

Figure 3: The KDD Process

10. Apply the data-transformations (pre-processing) done in step 6 to the target-segment as the mined pattern relies on it.

11. Select the best (ordered by the mined reponse-pattern) records (proportional to 80% of the budget) from the target segment of step 1.

12. Export the addresses of this selection and do the real mailing.

13. Compute a final evaluation, and store all the mailing-information (date, (non-) responses, segment, product, mined pattern, data-transformations, evaluation, ...) in the DWH/DWH-meta-data-Repository, such that it could be used as background knowledge for further actions.

Compared to the standard KDD-process (see fig 3), step 0 corresponds to data selection, collection integration and cleaning. We will not investigate that in detail, as this is well investigated in the data warehouse community, but instead rely on an already build data warehouse (DWH),

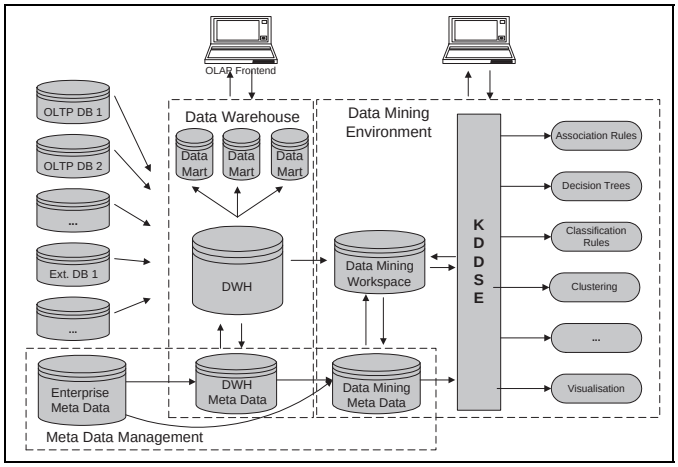

Figure 4: The KDD Environment

which is in fact an ideal first steps in setting up a KDD process [Inmon, 1996]. It is important to distinguish between data warehouses and data marts[3]. This difference is important for data mining, as data marts are not very useful for data mining, due to the application specific selection and aggregation of the fact data, i.e. they most often do not contain the data at the necessary level of detail needed for mining, e.g. the typical data mart for the monthly product sales (number, value, ...) by region, product and time, could not be used to analyze customer behaviour with mining methods, as it does not contain any reference to specific customers, even if it is an aggregation of customers buying behaviour[4]. This relation between the data warehouse, the data marts and the data mining environment is shown in fig 4.

The basic structure of our data warehouse as far as it is relevant for this business-case is shown in figure 2[5]. When we start the pre-processing phase on top of such a DWH, we are faced with

- a normalized, multi-relational database as data source, whereas most existing data mining algorithms are single-table based,
- many features and tables which are only partially relevant for the current business-case.
- a feature value coding optimized for maintenance (e.g. weekly update) of the data source, and not for optimal use within a specific data mining tool for a specific task.

For our mailing-case this means, that the mapping between DWH data representation and the end-users needed to specify a customer segment (step 2) is far from being trivial. So the first step (1) is to generate a more application oriented view on top of the DWH schema. As this is an application oriented view, it depends strongly on the business-case to solve (e.g. for product-development the base-level is not households, but insurance-contracts). Therefore, it already builds the first group of pre-processing steps to be handled (stored, retrieved and adapted) by our case-base. The other group is the one described in step 6. Whereas, step 1 does not need very much adaption for reuse, if we understand by reuse just the application of this schema to another target segment (i.e. to 2.b instead of 2.a), this is not true for step 6, as the different target segments have very different properties. Therefore quite different attributes are relevant for predicting the behaviour, e.g. as a general rule it could be stated that most households addressed by 2.a are not ensured so far, whereas the previous insurance behaviour is very likely to be important for predicting the response for segment 2.b.

3 A Case-Based Reasoning System for Pre-Processing

This section shows a first idea of how a possible system could look like. The system has two main parts. One is to support an end-user, who has to solve a specific mining task. The other is to describe and store a new mining task within the system. The idea is to develop a case base where special designed Mining Mart (MM) cases are stored for the mining supported business-cases regularly needed.

Figure 5 illustrates the use of the CBR-system. An end-user comes up with a specific business-task he want's to solve with data mining (e.g. doing the mailing action as described in the last section) and asks the system, if there exists a case for this task. If there is a case, that fits the users intention, he can use it, otherwise he has to contact a mining mart power-user to set up a new case.

To meet these requirements, a case must have two associated parts:

a) a business-case description for retrieval

[3] In the DWH-literature, these terms are mixed up, similar to KDD and data mining in the DM-literature. With DWH we mean an integrated and cleaned, but still relational and mostly normalized general purpose database, whereas the integrated and aggregated data for a specific OLAP application is called data mart.

[4] However, the hierarchical dimensions, e.g. for product, time and region, of the data mart could be very useful background knowledge for mining.

[5] Further information on this data warehouse, in particular a data extract, are available on our Web for further experiments. It can be obtained from http://research.swisslife.ch/kdd-sisyphus/.

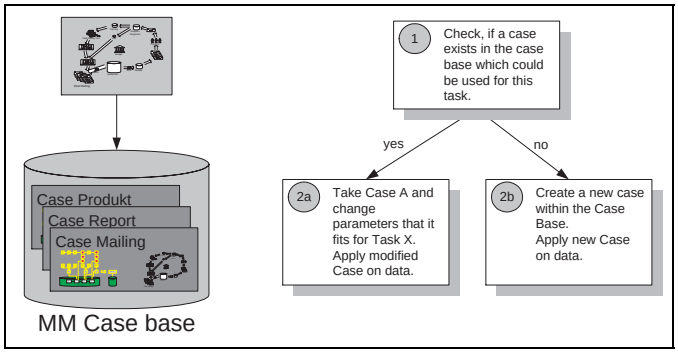

Figure 5: The CBR Approach

- no 'unknown' (NULL) values are allowed for specific attributes
- scalar and ordinal attributes have to be numeric
- nominal attributes must have character values or be represented as sets of boolean values
- no numeric or no non-numeric attributes are admissible
- not more than N different values are allowed for nominal attributes
- always the same scale for numeric attributes is required
- no key attributes are respected
- input data must consist of a single flat table

Figure 6: Input restrictions of data mining algorithms

b) a re-usable technical specification/implementation of the KDD-process for this business-case, e.g. the representation of the steps 2-12 of the mailing-case

Case-retrieval is not an issue of this paper, for the purpose of the paper it could be seen just as manually picking one from the list of cases[6], presented to the user by name or by a graphic like the one in Fig. 1.

Concerning part b of the case we will limit our focus in this paper on the representation and re-use of the pre-processing operations for data transformations, i.e. the pre-processing steps 1, 2, 3, 5 and 6, and the related data transformation operations 8 and 10 of the mailing-case.

We will neither cover the data mining (modeling) step itself (step 7 in the mailing-case) nor post-processing (testing/evaluation/use, steps 9 and 11) in this paper. However, at least clustering, classification learning and regression could be formalized as pre-processing operations constructing new attributes i.e. the new class, the predicted class and the predicted value (see sec. 3.1.2) within our framework.

Important is however that the selection of the data mining tool is part of the case. This frees the end-user from the difficult task of selecting a data mining tool (see sec. 4.2), and ensures that the goals of pre-processing:

1. to provide the most relevant data for a certain task,
2. to provide the data in a form most suitable for mining,
3. to fulfill the input restrictions of data mining tools (fig. 6 lists some typical restrictions), and
4. to generate useful and necessary background knowledge from meta-data

as far as influenced by the selection of the data mining tool remain the same for a selected case, i.e. that the pre-processing operations used in the case because of the selected data mining tool are still the needed ones.

However, as already discussed above (i.e. at the end of section 2), even just the selection of another target segment from the same DWH has a very strong influence on the pre-processing goal 1. We therefore need to adapt the case to reuse it successfully on the new data.

Another approach related to the combination of pre-processing and data mining within the KDD-process is multi-strategy learning (MSL) [Michalski, 1991; Michalski and Kaufman, 1998]. We follow this approach in our framework design so solve this problem of case-adaption to new data. In particular, we will combine constructive induction [Mehra *et al.*, 1989], with feature selection [Liu and Motoda, 1998b; Liu and Motoda, 1998a]. Additionally, several base feature construction operations are based on learning methods, e.g. discovering optimal discretizations and groupings. At a first sight this seems to be a very promising approach to total automation, as it does not need large amounts of currently unknown knowledge, but relies on systematic or heuristic exploration on the space of possible data transformations. Pre-processing and mining of real world data by current systems in this manner creates an additional problem. It is likely that even heuristic MSL approaches will get lost in the huge search space of possible data transformations, and the advantage of not needing domain knowledge becomes the disadvantage of not being able to use such domain knowledge to restrict the search space to a manageable size. Our case-based approach to integrate pre-processing and data mining can be seen as an attempt to set up an environment where available knowledge can be used to specify

[6] As long as we are limited to the mining based business-cases for the DWH of a single company or business-unit this will probably be sufficient. For the whole Mining Mart project there will be a workpackage concerned with business-case description for retrieval.

the some transformations and especially the structure of the case manually, and where integrated multi-strategy learning is used

- to automatically solve manageable sub-problems where such knowledge is not available, and
- to locally optimize the adaptation of the case to new data.

3.1 The Atoms of Pre-Processing Cases

The base-operations of pre-processing cases, investigated in this paper are sampling and segmentation, feature construction within a table and over related tables and feature selection[7]. In this approach feature construction and dropping of base-features is separated, as features may be needed for more than one feature construction operation (e.g. the date of birth of a person is needed to construct the age of the person, the entry-age into a contract and the end-age of a contract for every contract).

3.1.1 Sampling & Segmentation

Sampling and segmentation are used to reduce the number of data records within the training data. The underlying goals of these operations are the following:

- Improve speed and reduce memory requirements of the mining tool
- Focus on rare or special cases
- Rebalance the class distribution
- Specify a target group
- Use only clean data

These operations leave the number of attributes and the data records unchanged, however the statistical properties of the population may change dramatically. Sampling can only be applied to one data table.

Examples

- Random sampling or splitting
  Extracts data records out of the set of training data by random.
  Parameters: number of records to extract.
- Typical cases / Atypical cases
  Extracts data records which fulfill the (a)typical case within the training data.
  Parameters: Condition of the (a)typical case.
- Segmentation
  Extracts data records which fulfill a special segmentation-pattern.
  Parameters: Segmentation condition (e.g. 2.a and 2.b).
- Dataquality-motivated sampling
  Extracts data records which values have a certain degree of quality.
  Parameters: Qualify-condition (e.g. all records which have less than 2 unknown values).
- Rebalancing
  Noise-tolerant mining tools cannot be used to build a classification for very unbalanced class distributions (e.g. $95\% - 5\%$, with 5% being an optimistic estimate for responses of a mailing.), a sampling favoring the members of the minority class has to be applied first.
  Parameters: class distribution (e.g. $55\% - 45\%$) after rebalancing.

3.1.2 Attribute Construction within a relation

Simple attribute construction creates a new attribute within one data table or view. The new attribute is based on one or more base attributes and groups their values into a more general form. The number of data records in total is the same as before the operation. But when considering the base attributes replaced by the newly created one the number of distinguishable data records is possibly reduced (except for relativation and rescaling).

Goals

- Improve data-coding relative to the capabilities of the mining tool
- Create a new attribute which can be better used for the mining task

Examples

- Discretization
  Is applied to attribute(s) of the type ordinal or scalar and creates a new attribute of the type nominal ordered
  (ordinal/scalar $\rightarrow$ nominal ordered).
  Parameters: base attribute(s), output attribute,

[7] In [Morik, 2000] the pre-processing of time data is investigated, which will be part of the future Mining Mart, too.

number of created intervals for the output attribute (e.g. values of the income-attribute(s) are grouped into i0, i1..., i10).

- Grouping
  Is applied to attribute(s) of the type nominal which has/have no hierarchy and creates a new nominal unordered attribute.
  (nominal unordered → nominal unordered).
  Parameters: base attribute(s), output attribute, number of created groups, number of data records in one group (e.g. profession descriptions are grouped into groups of professions).

- Abstraction
  Is applied to attribute(s) of the type nominal or scalar and creates a new attribute of the type nominal ordered.(nominal, scalar → nominal ordered).
  Parameters: base attribute(s), output attribute, hierarchy, output of hierarchy level (e.g. looking at the household level instead of person level).

- Relativation
  Puts one attribute in relation to another attribute. This works only on numeric or date attributes and doesn't change the number of different data records.(numeric, date → numeric ordered).
  Parameters: base attributes, output attribute, operation between the base attributes (e.g. calculating the age from Sysdate and birthdate, calculating the quotient of income and premium sum).

- Cleaning
  Eliminates rare values of data records by creating a new attribute.(any type → any type).
  Parameters: base attributes, output attribute, which value of the base attribute shall be replaced by which new value (e.g. replacing the entry age of a person smaller than one by one).

- Unknown elimination
  Replaces unknown values with a specified new value.(any type → any type).
  Parameters: base attributes, output attribute, specified replacing value (e.g. replacing the unknown values of attribute age by the mean value or most frequent value).

- Scaling
  For all distance-based mining tools (e.g. clustering and instance based learning) the scale of the numeric attributes is very important, i.e. attributes with larger values are more influential on the result. To avoid this usually unintended weighting of attributes, all attributes have to be rescaled, e.g. to a fixed standard deviation or interval. (scalar→ scalar).
  Parameters: standard deviation or interval

3.1.3 Multi-Relational Attribute Construction

Joins and aggregations are used to put information from several related tables into one base table. To avoid unwanted changes of the target population distribution, the object identity within the base table has to stay unchanged (the number of different data records is still the same after the operations). To avoid the loss of base-table record-joins, outer-joins should generally be used, and to avoid the duplication of base-table records, simple joins must not be used for 1:N or N:M related tables (as this would duplicate records in the base table. E.g. it is not a good idea, to send duplicated mail to a household, for every person and contract involved). Instead the aggregation operations of this section have to be used[8].

Goals

- Fit the single table requirement of most DM-tools
- Reduce the complexity for ILP-DM-tools
- Avoid unwanted changes of the population distribution.

Examples

- Sum
  Creates a scalar attribute, e.g. premium sum of a product, income of a household.

- Min, Max
  Creates a scalar ordered attribute, e.g. smallest, highest premium sum of a product, smallest age of a member of the household.

[8]The design of these operations is in a way inspired by theoretical results in Inductive Logic Programming (ILP) on how to translate determinate hornclauses into propositional logic [Džeroski *et al.*, 1992; Kietz and Džeroski, 1994], which are also used in the ILP-system DINUS [Lavrač and Džeroski, 1994] for propositionalisation. Newer and more general propositionalisation approaches like [Alphonse and Rouveirol, 1999] are not used, as they are based on duplications of records in the base-table, which does not seem to be adequate in this context. Instead, we use operations inspired by Description Logics (DL) [Brachman and Schmolze, 1985] and the constructive induction operations of the DL-learning system KLUSTER [Kietz and Morik, 1994] to generate determinate features from indeterminate relations. These relations will also be further investigated in a future Mining Mart Workpackage.

- Count different
  Creates a numeric attribute, e.g. how many persons a household has, how many insurance contracts a household has.

- Count X = V
  Creates a numeric attribute, e.g. how many children a household has , how many different values an attribute has.

- Exist X = Y
  Creates a binary attribute, e.g. exists an insurance contract of type 3a for one household.

- All X = Y
  Creates a binary attribute, e.g. are all insurance contracts of a household of type 3a, are all persons of a household adults.

3.1.4 Attribute Selection

Attribute Selection drops attributes which should not be in the mining input and/or result, e.g as they are clearly uninteresting, not usable or difficult or expensive to measure for new data. The number of different data records and also the number of the total data records stays the same.

Goals

- Drop attributes which don't fit the input requirements of a DM-tool

- Drop attributes which are strongly dependent on or the base of attribute construction for other attributes

- Improve processing speed

- Guide the build-in attribute selection process of the DM-tool

Examples

- Feature / attribute selection
  Only the important attributes are chosen as input for the DM-tool. Selecting the attributes can be done manually, through input-restrictions associated with the mining tool or by feature selection methods [Liu and Motoda, 1998b; Liu and Motoda, 1998a].

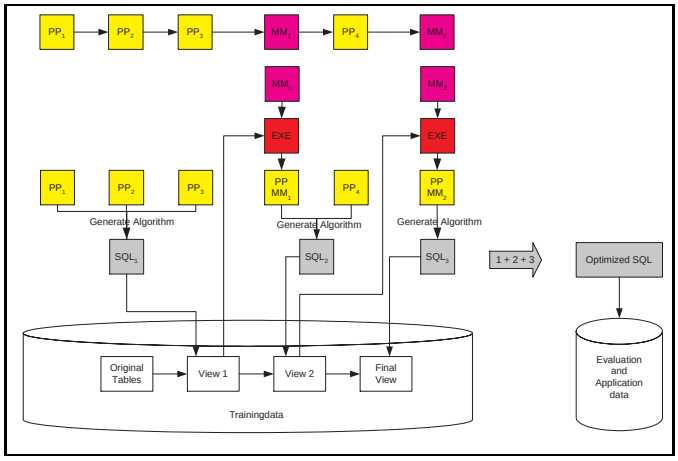

Figure 7: Chain of pre-processing operations

3.2 The Construction of Pre-Processing Cases

The development process contains several tasks. A chain[9] of several pre-processors and MSL tools, which have a defined order and specific parameters, has to be developed. In this chain MSL tools can also be used as preprocessors. The chain is locally optimized to supply the best mining result on training data. The whole chain of all pre-processing operations should be generated into one optimized SQL-statement, which can be executed on the application data (e.g. steps 8 and 10 of the mailing case).

During the process of developing an optimal preprocessing chain the mining mart power-user has to execute several iterations to find the best fitting (global optimized) chain of pre-processor operations and MSL tools. But even the power-user can gain from integrated MSL-tools as they help him to find a local optimized solution automatically. In a first attempt a pre-processor chain is defined and executed on the training data. Then the chain is changed and executed on the training data again. After all iterations are completed several pre-processor chains with mining results exist. The chain with the best result has to be chosen to be the best fitting one[10].

Figure 7 shows a complete chain of pre-processing operations and MSL tools and also how SQL-statements are generated from that. The chain contains different manual pre-processing operations (PP) and MSL tool supported pre-processing operations (MM) in a specific order. Usually the last element of an operation chain is the mining

[9] In the implementation do not use chains of operations, but hierarchical directed acyclic graphs.

[10] For future versions the other ones could be stored as variants in the case base which help the adaption of a case to a new target segment.

tool. The other elements of the chain can be either manual or MSL tool supported pre-processing operations. The MSL-tools are used to discover the parameters of the associated manual pre-processor operations. How many elements exists and which order they have depends on the task itself and on the needed input parameter of the next element. Manual pre-processing operations are based on SQL-code. MSL tools are stand-alone executables. For performance reasons several pre-processor operations can be translated into just one SQL-statement in case they are one after another.

The result of a SQL-statement is a view in the database which acts as an intermediate result, i.e. the view is needed as the input for a following MSL tool. The result of a MSL tool is used as the parameters of the associated SQL-code. As an example consider the discovery of a discretization as the result of a MSL tool. The output of the MSL tool is a mapping of intervals of the base-attribute (B) to nominal values of the new attribute (A), e.g. {if $B < 18$ then $A =$ child, if $18 \leq B$ then $A =$ adult }. This (and the old column) is used as a parameter of the SQL-function defining a new column in the next intermediate or the final view.

After the whole chain is executed, the final view and several SQL-statements exist. If it is possible, the SQL-statements can be merged and optimized to one statement. This statement or the single generated SQL-statements can also be executed on the testdata (Step 8 of the mailing-case) or the application data (Step 10 of the mailing-case).

4 Related Work

Precondition for the success of this framework is the existence of a user-friendly and efficient KDD environment. We intend to achieve this precondition through a KDD support environment (KDDSE, [Brachman and Anand, 1996]) that advances the state of the art of current KDDSEs (Section 4.1). Such an advanced KDDSE will utilize research results for semi-automatic process planning support as discussed in Section 4.2.

4.1 KDD Support Environments

The KDD-process itself is nicely defined in the CRISP-DM process model [Reinartz *et al.*, 1998]. The pre-processing phase is the most time consuming task for practical applications. Therefore we discuss the support given by KDDSEs in this process. Data mining environments that support pre-processing generally only support internal pre-processing of data already loaded into the KDDSE. It is problematic, if not impossible, to force the content of a data warehouse into the KDDSE before applying the first pre-processing operator. However, with the announcement of the SPSS Clementine-Server and the better integration of DB2 and IBM's Intelligent Miner this situation is currently changing. Another problem is, that the result of a pre-processing operation is often stored extensionally. This makes the reuse of operations difficult and furthermore, if a problem compounded of a series of pre-processing operations is executed, several nearly identical copies of the original data set must be stored to prevent the loss of the intermediate steps required for further documentation and re-application purposes.

As an alternative, most KDDSEs allow the use of manually specified SQL-expressions. However, specifying pre-processing operations in SQL is difficult, and the user is forced to select a sample of the database (via SQL) to be able to conduct the necessary pre-processing inside the KDDSE. The pre-processing environment to be developed in this project introduces support for in-database pre-processing operations. Particular research issues address the introduction of operations suited for multi-relational databases. The importance of support for multi-relational pre-processing is increasing with the introduction of commercial KDDSEs supporting multi-relational analysis (Kepler and Clementine after completion of the Aladin project). Multi-relational analysis is the next step in the evolution of KDDSEs towards allowing analysis of complex, large, and most importantly, relational company data warehouses.

4.2 Semi-automatic process planning support

Beginning with the MLT-Consultant [Sleeman *et al.*, 1989] there was the idea of having a knowledge based system supporting the selection of a machine learning method for an application. The MLT-Consultant succeeded in differentiating the nine MLT learning methods with respect to specific syntactic properties of the input and output languages of the methods. However, there was little success in describing and differentiating the methods on an application level that went beyond the well known classification of machine learning systems into classification learning, rule learning, clustering, and sloppy modeling. Also, the STATLOG ESPRIT-Project [Michie *et al.*, 1994], which systematically applied classification learning systems to various domains, did not succeed in estab-

lishing criteria for the selection of the best classification learning system. It was concluded that some systems have generally acceptable performance; and in order to select the best system for a certain purpose, they must each be applied to the task and the best be selected through a test-method such as cross-validation. Theusinger and Lindner [1998] are in the process of re-applying this old idea of searching for statistical dataset characteristics necessary for the successful applications of DM-tools. An even more demanding approach was started by Engels [1997]. This approach not only attempted to support the selection of DM-tools, but built a knowledge-based process planning support for the entire KDD-process. Until today this work has not led to an usable system [Engels *et al.*, 1997]. The European project MetaL now aims at learning how to combine learning algorithms and datasets. We do not believe that this top-down knowledge-based approach will lead to an usable environment in the short run, as it requires a large amount of very application-specific knowledge. Furthermore, it is widely agreed upon that even the manual KDD-process cannot be planned ahead of time in detail. The utility of an operation can often only be determined after a large number of further operations is executed. It is apparent that not enough knowledge is available to propose the correct combination of pre-processing operations. However, it is possible to collect knowledge to exclude illegal, meaningless and unsuccessful combinations of operations. Therefore, we propose to use case-based semi-automatic process planning. Our goal is a system which supports a group of experienced data mining and domain experts in creating initial cases of the KDD-process for a specific application (e.g. the mailing action) and a specific type of data (e.g. a company's data warehouse). The system then offers these cases to domain experts, and supports them in repeating the KDD-process on new data of the same type (e.g. the same data warehouse, updated with new data, the result of the last mailing, etc.) for a similar application (the next mailing action). Hence, only the first use/creation of a case will require substantial effort and DM-experience, whereas all further uses of this case will be much quicker and more inexpensive. Additionally, a larger number of data mining applications can be executed with a limited number of available DM-experts. This case-base may even be useful to acquire knowledge about the KDD-process itself, e.g. by the Meta-Level Learning Methods developed in MetaL. This information could be utilized for more sophisticated process-planning support of initial cases for new applications.

5 Conclusion

The Mining Mart framework described in this paper builds on the insight that current approaches for achieving the objectives described above tend to ignore theoretical results, which have been proven that no algorithm can claim to be systematically better than any other on every problem [Wolpert and Macready, 1995], and that nobody has yet been able to identify reliable rules for predicting, that one algorithm should be superior to others, i.e. a total automation of the KDD-process is not possible.

A constraint based graphical user interface based on the KDDSE Kepler utilizing meta-data shall guide users through the knowledge discovery task. The highest possible degree of automation for this process will be the aim of this project. However, as reasoned above, it cannot be expected that the user simply asks a high level question and selects a data set to be analyzed and everything else is done automatically. In particular, the task of proper transformation of the given data into a format that can be successfully analyzed by the available algorithms is difficult. As discussed above, testing of all possible approaches through pure multi-strategy learning is currently not practical because the required computational power is not accessible for any single user. However, user that have access to the Mining Mart can search the case-base for suitable solutions to their task at hand. If no proper solution is found, the task will be posted as a new challenge to the knowledge discovery experts.

The main innovation of this project will be the deep integration of the different research directions currently accessible only to experts into an uniform environment usable also by data mining non-experts.

Acknowledgements: This work has been partially found by the Swiss Government under the contract-no "BBW Nr.99.0158" as part of the European commision Research Project IST-1999-11993 (Mining Mart). We thank Cèline Rouveirol and Ulrich Reimer for very helpful comments on a draft of this paper and all partners of the mining mart project with whom we had fruitful discussions about the framework described in this paper.

References

[Alphonse and Rouveirol, 1999] E. Alphonse und C. Rouveirol. Selective propositionalization for relational learning. In *Proceedings of the Third European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'99)*. Springer Verlag, 1999.

[Brachman and Anand, 1996] R. Brachman und T. Anand. The process of knowledge discovery in database. In *Advances in Knowledge Discovery and Data Mining*. AAAI Press, 1996.

[Brachman and Schmolze, 1985] R. J. Brachman und J. G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171 – 216, 1985.

[Džeroski *et al.*, 1992] S. Džeroski, S. H. Muggleton, und S. Russell. PAC-learnability of determinate logic programs. In *Proc. Fifth ACM Workshop on Computational Learning theory*, pages 128–135. ACM Press, New York, 1992.

[Engels, 1997] R. Engels. Planning tasks for knowledge discovery in databases; performing task-oriented user-guidance. In D. Heckerman, H. Mannila, D. Pregibon, und R. Uthurusamy, editors, *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*, page 170. AAAI Press, 1997.

[Engels *et al.*, 1997] R. Engels, G. Lindner, und R. Studer. A guided tour through the data mining jungle. In D. Heckerman, H. Mannila, D. Pregibon, und R. Uthurusamy, editors, *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*, page 163. AAAI Press, 1997.

[Inmon, 1996] W. H. Inmon. The data warehouse and data mining. *Communications of the ACM*, 39(11):49–50, November 1996.

[Kietz and Džeroski, 1994] J.-U. Kietz und S. Džeroski. Inductive logic programming and learnability. *SIGART Bulletin*, 5(1), 1994.

[Kietz and Morik, 1994] J.-U. Kietz und K. Morik. A polynomial approach to the constructive induction of structural knowledge. *Machine Learning*, 14(2):193–217, 1994.

[Lavrač and Džeroski, 1994] N. Lavrač und S. Džeroski. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, Chichester, England, 1994.

[Ling and Li, 1998] C. X. Ling und C. Li. Data mining for direct marketing: Problems and solutions. In R. Agrawal und P. Stolorz, editors, *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 73–79. AAAI Press, 1998.

[Liu and Motoda, 1998a] H. Liu und H. Motoda. *Feature Extraction Construction and Selection, A Data Mining Perspective*. Kluwer Academic Publishers, 1998.

[Liu and Motoda, 1998b] H. Liu und H. Motoda. *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, 1998.

[Mehra *et al.*, 1989] P. Mehra, L. Rendell, und B. Wah. Principled constructive induction on decision trees. In *Proc. of the Eleventh Int. Joint Conf. on AI (IJCAI-89)*, pages 651–656. Morgan Kaufman, Los Altos, California, 1989.

[Michalski, 1991] R. S. Michalski. Inferential learning theory as a basis for multistrategy task-adaptive learning. In *Proceedings of the first International Workshop on Multistrategy Learning*, pages 3 – 18. George Mason University, 1991.

[Michalski and Kaufman, 1998] R. Michalski und K. Kaufman. Data mining and knowledge discovery: A review of issues and a multistrategy approach. In R. Michalski, I. Bratko, und M. Kubat, editors, *Machine Learning and Data Mining Meathods and Applications*. John Wiley & Sons LTD, Chichester, England, 1998.

[Michie *et al.*, 1994] D. Michie, D. Spiegelhalter, und C. Taylor, editors. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood, Chichester, England, 1994.

[Morik, 2000] K. Morik. The representation race – preprocessing for handling time phenomena. In *Proc. of the European Conference on Machine Learning, ECML-2000*. LNAI, Springer Verlag, 2000.

[Pyle, 1999] D. Pyle. *Data Preparation for Data Mining*. Morgan Kaufmann Publishers, San Francisco, California, 1999.

[Reinartz *et al.*, 1998] T. Reinartz, R. Wirth, R. Clinton, T. Khabaza, J. Hejlesen, und P. Chapman. The current crisp-dm process model for data mining. In F. Wysotzki, P. Geibel, und K. Sch"adler, editors, *Beitr"age zum Treffen der GI-Fachgruppe 1.1.3 Machinelles Lernen (FGML-98)*. Technical Report 98/11, Technical University Berlin, 1998.

[Sleeman *et al.*, 1989] D. Sleeman, R. Oehlman, und R. Davidge. Specification of consultant-0 and a comparision of several learning algorithms. Mlt-deliverable d5.1, Machine Learning Toolbox Esprit Project P2154, 1989.

[Staudt *et al.*, 1998] M. Staudt, J.-U. Kietz, und U. Reimer. A data mining support environment and its application on insurance data. In R. Agrawal

und P. Stolorz, editors, *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 105–111. AAAI Press, 1998.

[Staudt *et al.*, 1999a] M. Staudt, A. Vaduva, und T. Vetterli. Metadata management and data warehousing. Technical report, Information Systems Research, SwissLife, 1999. http://research.swisslife.ch/Papers/papers.htm.

[Staudt *et al.*, 1999b] M. Staudt, A. Vaduva, und T. Vetterli. The role of metadata for data warehousing. Technical report, Information Systems Research, SwissLife, 1999. http://research.swisslife.ch/Papers/papers.htm.

[Theusinger and Lindner, 1998] C. Theusinger und G. Lindner. Benutzerunterstützung eines kdd-prozesses anhand von datencharackteristiken. In F. Wysotzki, P. Geibel, und K. Schädler, editors, *Beiträge zum Treffen der GI-Fachgruppe 1.1.3 Machinelles Lernen (FGML-98)*. Technical Report 98/11, Technical University Berlin, 1998.

[Wolpert and Macready, 1995] D. Wolpert und W. Macready. No free lunch theorems for search. Techinal Report SFI-TR-95-02-010, Santa F Institute, Santa F, CA., 1995.